Optimasi Kinerja *Point of Sale (PoS)* dengan Penerapan Sinkronisasi Database Menggunakan *Middleware*

Ragilliyandi Erick Putra I*, Berlian Maulidya Izzati, Fitriyana Dewi*****,**
Jurusan Sistem Informasi, Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember (ITS) – Surabaya, 60111, Indonesia
erick15@mhs.is.its.ac.id* , berlian.maulidya15@mhs.is.its.ac.id** , fitriyana15@mhs.is.its.ac.id***

ABSTRACT

In the era of globalization such as today, many companies and organizations using WAN and cloud service technology to face the business world which is growth very rapidly in order to have a competitive advantage. One thing to note in the technology of cloud service is the efficiency and consistency of the data. The research objective to implement application of WAN and cloud service on the application of point of sale (PoS) at RAIN who has 10 branches in Surabaya. The problem faces by RAIN is the lack of network infrastructure provided by the internet mall where not able to support the real time activities and data of sales. This research proposes a solution by implementing a synchronization algorithm as a middleware application technology to address the insufficiency of the WAN network. The application serves as temporary storage when the network is offline, and the application could be the right solution to improve and support WAN shortage.

Keyword: database synchronization, point of sale, middleware

ABSTRAK

Pada era globalisasi seperti saat ini, perusahaan dan organisasi banyak menggunakan WAN (*Wide Area Network*) serta teknologi *cloud service* untuk mengimbangi dunia bisnis yang berkembang sangat pesat supaya memiliki keunggulan kompetitif. Salah satu hal yang perlu diperhatikan dalam teknologi *cloud service* adalah terciptanya efisiensi waktu dan konsistensi data. Objek kasus dalam penelitian ini yaitu penerapan penggunaan WAN dan *cloud service* dalam pemanfaatan aplikasi *point of sale* (PoS) pada sebuah bisnis butik bernama RAIN yang memiliki 10 cabang di kota Surabaya. Permasalahan yang dihadapai oleh RAIN adalah infrastruktur jaringan yang disediakan oleh pihak *mall* dirasa kurang memadai untuk penerapan database terpusat, seperti jaringan kurang stabil dan berdampak pada aplikasi PoS yang tidak dapat beroperasi. Sehingga, kegiatan jual beli pada butik terganggu. Oleh karena itu, RAIN menerapkan database terdistribusi pada masing-masing cabang, akan tetapi masih terdapat permasalahan lain seperti sinkronisasi database yang dilakukan secara manual. Penelitian ini mengusulkan solusi berupa teknologi *middleware* yang digunakan untuk mengatasi permasalahan tersebut. Aplikasi tersebut berisi algoritma sinkronisasi yang dapat bekerja saat jaringan sedang *online* maupun *offline*. Sehingga, aplikasi tersebut dapat menjadi solusi yang tepat untuk meningkatkan produktivitas dan efektivitas kerja butik RAIN.

***Kata kunci*:** sinkronisasi database, *point of sale, middleware*

1. Pendahuluan

Pada era globalisasi seperti saat ini, dunia bisnis telah berkembang dengan sangat pesat dan menjadi semakin kompetitif. Seiring dengan berkembangnya dunia bisnis tersebut, teknologi informasi pun dituntut untuk berkembang dan berinovasi agar dapat mengimbangi kemajuan dunia bisnis. Perusahaan dan organisasi akan semakin bergantung pada teknologi informasi yang paling efektif dan efisien dalam menjalankan bisnis mereka termasuk di dalamnya proses penyimpanan dan pengolahan informasi bisnis. WAN (*Wide Area Network*) menjadi salah satu teknologi pendukung kerja operasional perusahaan yang terdistribusi pada beberapa area geografis yang terpisah. Sebuah perusahaan besar biasanya memiliki lokasi kantor pusat yang berada di kota besar dengan beberapa kantor cabang di kota-kota lainnya. Sehingga tidak dapat dipungkiri lagi bahwa sebuah perusahaan pasti membutuhkan fasilitas penghubung dengan memanfaatakan teknologi WAN. Seiring dengan berkembangnya bisnis

perusahaan seperti peningkatan intensitas operasi dan tuntutan terhadap peningkatan produktivitas, para pemangku kepentingan perusahaan biasanya akan dituntut untuk bekerja dengan sistem yang lebih terkomputerisasi. Dimana sebisa mungkin semua proses kerja pada perusahaan dilakukan secara otomatis dengan penyimpanan data secara digital. Hal tersebut tentu akan berdampak pada kemudahan dan kecepatan dalam mengakses data yang dibutuhkan selama bekerja. Saat ini, penggunaan WAN yang diikutsertakan dengan pemanfaatan *cloud service* telah menjadi model perkembangan teknologi yang sudah tidak awam lagi untuk digunakan. Beberapa keuntungan menggunakan *cloud services* yaitu kemudahan dalam melakukan akses sistem, menawarkan skalabilitas, dan '*adaptable computing system*' untuk berbagai aplikasi yang berbeda sehingga mampu mengoptimalkan kapasitas dan biaya operasional dari infrastruktur yang digunakan [1]. Penyebaran dari teknologi *cloud computing* telah berkembang sebesar 14% pada tahun 2015 dengan total mencapai US $175 miliar [2].

Sinkronisasi pada dunia IT berkaitan dengan terciptanya sebuah konsistensi pada data yang digunakan. Data yang digunakan oleh *client* harus sama dengan data yang ada pada *server*, begitu juga sebaliknya, data yang ada pada *server* harus sama dengan data yang diolah oleh *client*. Sinkronisasi harus mengacu pada dua konsep yaitu sinkronisasi data dan sinkronisasi proses [3]. Untuk mencapai kesepakatan atau *commit* pada aktivitas tertentu, maka perlu adanya sebuah proses yang mampu menggabungkan aktivitas-aktivitas pengolahan data tersebut pada sebuah titik tertentu. Sinkronisasi data juga terkait dengan menjaga integritas data atau menjaga beberapa salinan/*backup* dari banyak dataset yang terkait satu sama lain [4]. Pada kenyataannya saat ini, tidak semua pemanfaatan WAN berjalan sesuai dengan rencananya. WAN sebagai media komunikasi yang menghubungkan setidaknya antara dua lokasi kerja yang terpisah dilintasi oleh berbagai jenis data dengan frekuensi pengiriman, besar data, protocol, dan waktu tenggang yang berbeda-beda. Sehingga dapat dikatakan bahwa paket yang beragam tersebut akan berebut tempat dan antri untuk melewati jaringan yang digunakan. Selain itu secara teknis terdapat banyak hal yang terjadi pada saat pengiriman paket, misalnya saja permasalahan mengenai terjadinya *latency* (waktu yang dibutuhkan untuk melintasi jaringan) dan *delay* (waktu yang dibutuhkan saat node pengirim mengirimkan data hingga node penerima menerima data yang dikirimkan), atau bahkan tidak adanya koneksi internet yang seharusnya menghubungkan *server* dan *client* [1]. Maka dari itu dibutuhkan sebuah optimasi untuk mengatasi permasalahan yang terjadi dengan menggabungkan WAN sebagai bagian dari infrastruktur *cloud service*.

Objek kasus dalam penelitian ini yaitu penerapan penggunaan WAN dan *cloud service* dalam pemanfaatan aplikasi *point of sale* (PoS). Studi kasus yang diambil dalam penulisan ini adalah permasalahan yang terjadi pada butik RAIN dalam pemanfaatan aplikasi PoS. RAIN merupakan butik yang menjual beraneka ragam pakaian dan aksesoris khusus wanita yang memiliki gudang dan sepuluh cabang yang lokasinya tersebar di beberapa *mall* yang ada di kota Surabaya. RAIN menggunakan infrastruktur jaringan internet yang disediakan pihak *mall* dalam memanfaatkan teknologi WAN agar masing-masing cabang dan gudangnya dapat saling terhubung. Namun penggunaan infrastruktur jaringan internet yang disediakan oleh pihak *mall* kurang memadai untuk pemanfaatan aplikasi PoS dengan database terpusat, karena pemilik bisnis tidak dapat berbuat lebih ketika jaringan sedang *offline* dan berakibat tidak dapat beropersainya aplikasi PoS. Sehingga pemilik bisnis menggunakan database terdistribusi pada masing-masing cabang. Akan tetapi, terdapat permasalahan lain seperti sinkronisasi data yang dilakukan secara manual dan akses data antar cabang yang membutuhkan waktu lama. Proses sinkronisasi yang digunakan saat ini menggunakan beberapa teknik, diantaranya yaitu pemanfaatan *email* sebagai salah satu cara dalam melakukan sinkronisasi database, data transaksi harian diekspor dan dikirim secara manual melalui *email* dari setiap cabang butik kepada kantor pusat pada akhir jam kerja. Setelah data diterima oleh kantor pusat, maka kantor pusat akan memasukan data tersebut pada database pusat. Apabila terjadi perubahan pada database pusat, seperti perubahan harga atau penambahan barang, maka kantor pusat akan mengekspor data tersebut dan mengirimkannya secara manual melalui email kepada setiap cabang butik untuk dilakukan update pada database-nya. Hal ini dilakukan oleh setiap cabang butik secara-terus menerus pada setiap akhir jam kerja dan cara ini dinilai tidak efektif dalam segi efisiensi waktu dan profesionalisme kerja. Selain itu, terdapat permasalahan lain dalam akses database antar cabang. Salah satunya untuk melakukan pengecekan ketersediaan barang secara *realtime* harus terhubung langsung dengan database cabang, karena database yang ada pada kantor pusat adalah data transaksi hari sebelumnya, dan hal tersebut hanya memungkinkan apabila jaringan pada cabang sedang *online*.

Penelitian ini mengusulkan solusi berupa teknologi *middleware* yang digunakan untuk mengatasi kekurangan dalam pemanfaatan jaringan WAN. Aplikasi tersebut diharapkan dapat menjadi solusi yang tepat untuk meningkatkan produktivitas dan efektivitas kerja butik RAIN. Kelebihan dari aplikasi tersebut adalah kemampuannya yang dapat diakses kapan saja (dalam kondisi *online* maupun *offline*) tanpa adanya gangguan dari koneksi yang dialami. Terkadang pengguna yang berada di cabang tidak dapat melakukan *update* database pada *server* pusat karena beberapa kendala yang terjadi, misalnya saja terjadi kegagalan pada *server*, *server down*, ataupun tidak ada koneksi internet. Maka secara otomatis semua data yang digunakan oleh pengguna yang terdapat di cabang akan tersimpan di dalam sistem lokal mereka sendiri. Namun nantinya data ini akan ditransfer secara otomatis dari sistem lokal mereka ke *server* pusat ketika tersedia koneksi internet (*online*).

2. Penelitian Sebelumnya

Dalam bagian ini akan dibahas mengenai jurnal acuan yang digunakan dalam penelitian ini. Terdapat beberapa jurnal utama yang menjadi acuan dalam penelitian ini, jurnal tersebut merupakan jurnal yang dipublikasikan dalam situs jurnal akademik dari tahun 2012 hingga tahun 2015 dengan penjelasan sebagai berikut:

Isak Shabani, dkk. (2012) dengan penelitian yang berjudul "*Solving Problems in Software Applications through Data Synchronization in Case of Absence of the Network*". Dalam penelitian ini, Isak Shabani, dkk. Mengambil topik permasalahan mengenai sistem informasi absensi mahasiswa. Peneliti mengajukan algoritma sinkronisasi berbasis *web service*, dimana dengan algoritma tersebut *software* dapat berjalan dengan baik tanpa dipengaruhi oleh kondisi jaringan yang sedang *online* maupun *offline*. Ketika terjadi kendala dalam penggunaan sistem informasi absensi mahasiswa, maka pengaturan akan langung berlalih ke mode *'offline'* sehingga data akan tersimpan ke dalam database lokal, dan ketika komponen sinkronisasi (*proxy client*) telah mendeteksi adanya *network* maka komponen tersebut akan memanggil *web service* yang akan mengirimkan data untuk disinkronisasi dengan *server* pusat [5]. Sehingga penggunaan sistem informasi tersebut menjadi lebih produktif tanpa dipengaruhi oleh kendala yang biasa terjadi. Penelitian lain dilakukan oleh Muhsin Shodiq, dkk. (2015) yang berjudul "*Implementation of Data Synchronization with Data Marker using Web Service Data*". Muhsin Shodiq, et al. menerapkan sinkronisasi data *bidirectional* dan *unidirectional* menggunakan *data marker* dan *web service*. Dalam penelitian tersebut terdapat permasalahan bahwa data yang ada pada database *server* belum tentu merupakan data yang paling baru, hal ini disebabkan karena setiap pengguna dapat melakukan *update* dari perangkat yang berbeda. Untuk menyelesaikan permasalahan tersebut, peneliti menggunakan *marker* pada masing-masing data. *Marker* yang dimaksud berupa *timestamp* dari waktu terakhir sinkronisasi, waktu pembuatan data, dan waktu terakhir modifikasi data. Dengan *marker* tersebut *server* dapat menentukan data mana yang paling baru, dan dari sisi *client* dapat menghindari permintaan data berulang. Sehingga tidak akan terdapat perbedaan data yang diakses oleh pengguna pada setiap perangkat yang pengguna gunakan meskipun menggunakan perangkat yang berbeda [3].

Sedangkan penelitian mengenai pemanfaatan sinkronisasi untuk media penyimpanan dilakukan oleh Yusuf Sutanto, dkk. (2015) yang berjudul "*Inventory Management Optimization Model with Database Synchronization through Internet Network (A Simulation Study)*", peneliti melakukan penelitian mengenai perbedaan penggunaan *synchronization database* dan model klasik yaitu *physical internet model* serta bagaimana penerapannya pada *database inventory* untuk mengurangi biaya operasi pengiriman barang. Konsep *physical internet model* yaitu dengan menghubungkan berbagi jaringan komputer logistik yang independen dan menggunakan jaringan tersebut sebagai *open logistic network* sehingga nantinya perusahaan akan memanfaatkannya sebagai media untuk mengetahui mengenai sisa barang atau permintaan barang dari ritel secara manual. Bebeda ketika menggunakan *database synchronization*, pengumpulan data barang dilakukan secara terpusat dengan melakukan replikasi data dari masing-masing *retail* ke *server* pusat. Selanjutnya data tersebut diproses sehingga menghasilkan daftar barang yang habis dan rincian permintaan yang dibutuhkan oleh setiap *retail*. Selanjutnya daftar tersebut akan dikirimkan ke pemasok melalui jaringan internet sehingga ketika pemasok penerima daftar tersebut, maka distribusi akan langsung dilakukan sesuai dengan urutan lokasi *retail*. Disamping itu tujuan utama dari penelitian ini adalah memberikan alternatif manfaat dari model yang diajukan sehingga didapatkan manfaat yang dibutuhkan dan diinginkan ketika menggunakan model tersebut [6].

Jue Wang, dkk. (2015) dengan penelitian yang berjudul "*Research and Design of Distributed Database Synchronization System Based on Middleware*" membahas mengenai konsistensi data pada database terdistribusi yang berbasis model *peer-to-peer*. Sinkronisasi database tersebut menggunakan *middleware* yang memiliki berbagai modul yaitu *transaction management module, collision detection module, communication module,* dan *transaction execution module. Middleware* tersebut mampu mengurangi waktu respon saat melakukan transaksi, mengurangi lalu lintas yang tidak diperlukan dan memperbaiki performa sistem. Hal yang sangat diperhatikan dalam penulisan ini adalah keberadaan konsistensi banyak data. Sinkronisasi database terdistribusi menjadi salah satu solusi dalam transaksi yang dilakukan secara bersamaan. Database yang diterapkan dalam penelitian ini dibagi menjadi dua yaitu *database synchronization middleware* dan *database system*. Pada *database synchronization middleware*, transaksi data akan di-monitoring dan dianalisis secara *real-time* sehingga konsistensi data akan tetap terjaga. Sedangkan yang dimaksud dengan *database system* adalah database lokal yang tersedia di masing-masing node atau cabang, atau dapat dikatakan database lokal merupakan *remote database* [7].

3. Sinkronisasi Data

Menurut Aashima dan Anit Kaur [4], sinkronisasi data merupakan proses pembentukan konsistensi data antara database utama sebagai target penyimpanan data dan database cabang sebagai target tujuan atau sebaliknya, sehingga terbentuk konsistensi data dari waktu ke waktu antar keduanya. Dalam proses sinkronisasi, data urutan aktivitas tertentu yang terjadi didasarkan pada urutan dan ketentuan yang berlaku sehingga integritas data akan tetap terjaga [4]. Menurut Naveen Malhotra dan Anjali, dengan konsep yang jelas namun berbeda dengan pendapat sebelumnya mengatakan bahwa sinkronisasi data merupakan kebutuhan untuk menyimpan beberapa salinan dari satu set data yang terkait satu sama lain [8], artinya sinkronisasi data merupakan proses pembentukan konsistensi antara data dari sumber penyimpanan kepada sebuah target penyimpanan yang lain maupun sebaliknya, sehingga

tercipta harmonisasi data secara terus menerus dari waktu ke waktu. Hal serupa juga diungkapkan dalam penelitian yang dilakukan oleh Yi Lin dan Bettina Kemme yang menyatakan bahwa sinkronisasi database merujuk pada sebuah sistem yang memiliki dua atau lebih database dengan data tersebar dari satu database ke database lainnya. Seluruh database harus konsistensi antara satu database dengan database lainnya. Yang dimaksud konsistensi adalah jika terdapat perubahan data dalam satu database, maka perubahan tersebut juga harus terjadi pada database lainnya [9] [10].

Sinkronisasi biasanya dibutuhkan ketika beberapa orang dalam sebuah organisasi atau beberapa cabang perusahaan yang berada pada daerah operasi yang berbeda atau jauh dari pusat sehingga tidak memiliki akses ke database pusat secara langsung. Padahal dalam melakukan kegiatan operasional sehari-hari pekerjaan mereka membutuhkan aktivitas '*share information*' antar cabang atau antara cabang dan pusat maupun beberapa orang dalam perusahaan. Sehingga salah satu solusi yang dapat dilakukan untuk melakukan komunikasi antar database yaitu dengan menggunakan teknologi jaringan *peer to peer*. Perubahan yang dilakukan biasanya tergantung pada kebutuhan setiap komputer untuk memodifikasi versi data asli melalui proses impelementasi dalam sistem terdistribusi. Selanjutnya data akan disinkronisasi oleh aplikasi penghubung atau *middleware* yang ada pada sistem tersebut sehingga pertukaran pembaharuan data dapat dilakukan.

4. *Middleware*

Middleware dapat diartikan sebagai sebuah *software* yang terletak di antara aplikasi dan sistem operasi, jaringan atau database. *Middleware* memudahkan para pengguna dengan cara menyembunyikan detail database dan jaringan dari aplikasi yang terdistribusi. Seiring dengan berkembangnya sistem komputasi dengan bermacam-macam teknologi yang ada, *middleware* dapat menjadi salah satu fasilitas penghubung dengan cara melakukan "*virtual homogenity*" di dalam sistem tersebut. Saat ini, *middleware* juga dikembangkan dalam bentuk *portable* untuk berbagai macam *platform* sehingga *middleware* dapat berperan untuk menyamakan tampilan dan layanan dimanapun aplikasi tersebut dieksekusi. Terdapat dua bentuk dukungan komunikasi di dalam *middleware* yaitu *Remote Procedure Call (RPC)* yang mencakup data transfer, *network programming*, dan *failures*. Sedangkan bentuk komunikasi yang kedua adalah dalam bentuk *messaging*. Layanan yang diberikan oleh *middleware* mencakup 8 layanan adalah: *1) directory services 2) transaction services 3) security services 4) management services 5) event services 6) persistence services 7) load balancing 8) configuration services* [11].

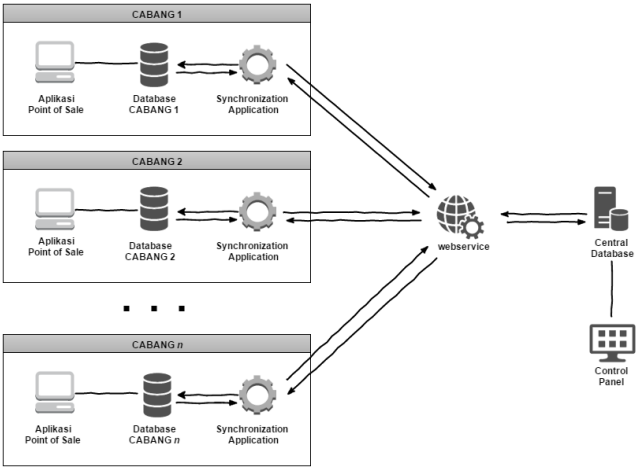
Layanan *middleware* yang digunakan dalam penelitian ini yaitu *sycn app* yang berperan sebagai salah satu pendukung dalam melakukan interaksi dengan jaringan pada aplikasi PoS untuk mengetahui *last sycn* lokal database dengan menggunakan *timestamp* yang didapatkan dari aplikasi PoS dan melakukan pendeteksian jenis data yang dikirimkan. Nantinya dilakukan proses lebih lanjut dengan menggunakan *web service* untuk melakukan *sync database* pusat dengan metode yang sama.

Web service merupakan suatu teknik terstandar dengan tujuan untuk menyampaikan sebuah layanan pada jaringan terdistribusi. Berbagai sumber menggunakan definisi berbeda mengenai teknologi *web service*, namun secara umum *web service* memiliki karakteristik sebagai berikut [12]:

1. *Web service* biasanya dapat diakses menggunakan *protocol* internet yang sudah umum diketahui seperti HTTP yang memungkinkan berbagai sistem yang berbeda dapat mengaksesnya.
2. *Web service* mendukung gabungan dari berbagai arsitektur yang berbasis pada layanan. Layanan tersebut menyediakan sebuah informasi yang dihubungkan dengan sebuah direktori khusus. Contohnya saja adalah bahasa pemrograman dan sistem operasi yang tersembunyi dari para penggunanya.
3. *Web service* mengirim dan menerima kode pesan XML yang digunakan untuk komunikasi dan membantu menjembatani antara sistem-sistem yang memiliki komponen, model, sistem operasi dan bahasa pemrograman yang berbeda

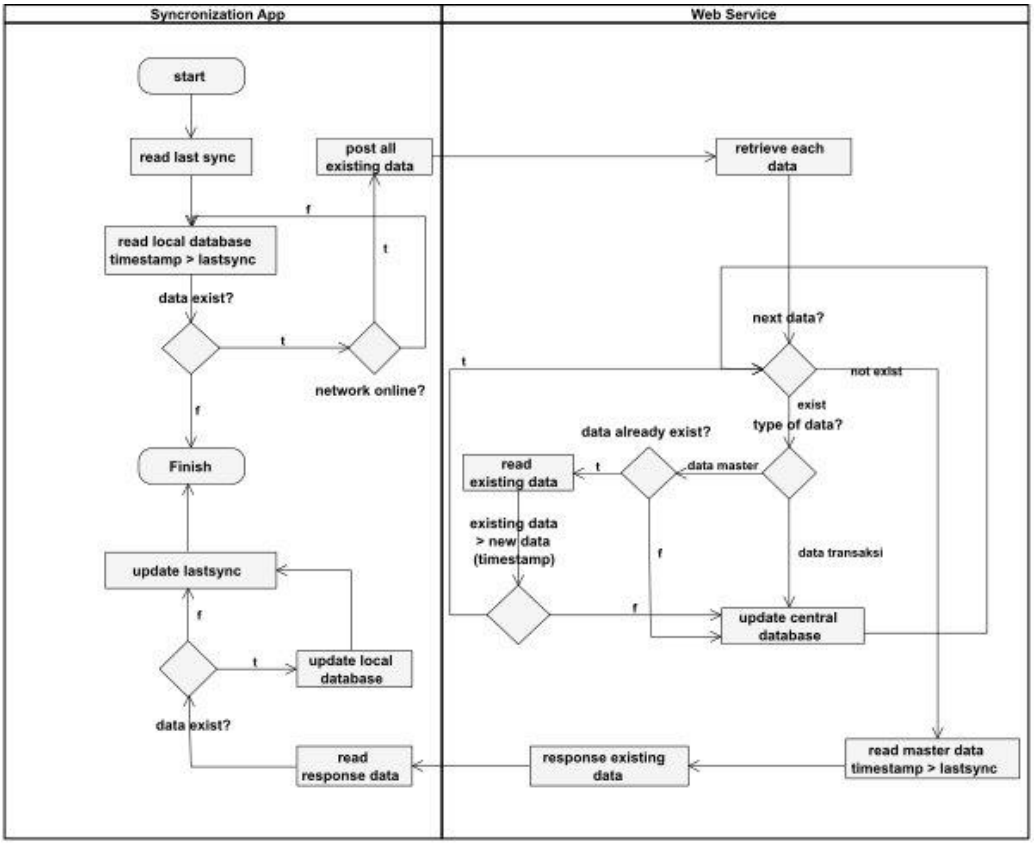
Fitur utama dari *web service* adalah kemampuannya menyediakan layanan secara timbal balik dan terpisah dari *platform* manapun. Teknologi *web service* tidak memberikan standarisasi pada *platform* apa saja yang bisa mengeksekusi, namun hanya sebatas timbal balik apa saja yang dibutuhkan [12]. Sekalipun mirip dengan *Application Programming Interface* (API) berbasis *web*, *web service* lebih unggul karena dapat diakses dari jarak jauh melalui internet. Pemanggilan *web service* bisa menggunakan bahasa pemrograman apa saja dan dengan *platform* apa saja, sementara API hanya bisa digunakan dalam *platform* tertentu. *Web service* dapat dipahami sebagai *Remote Procedure Call (RPC)* yang mampu memproses fungsi-fungsi yang didefinisikan pada sebuah aplikasi *web* dan menampilkan sebuah API atau *User Interface* (UI) melalui *web*. Kelebihan *web service* adalah: 1) lintas *platform*, 2) tidak tergantung dengan Bahasa pemrograman manapun, 3) mampu menjadi jembatan penghubung antar database tanpa perlu *driver* database dan tidak harus mengetahui jenis DBMS, serta 4) mempermudah proses pertukaran data [2].

5. Metodologi



Gambar 1. Struktur Desain Sistem

Penelitian ini menggunakan *syncronization app* dan *web service* sebagai *middleware* dalam proses sinkronisasi database. *Middleware* berperan sebagai perantara dalam sinkronisasi data antara database cabang dengan database pusat. Data dalam database RAIN dibagi menjadi dua jenis, yaitu data master dan data transaksi. Data master meliputi data *customer*, data *supplier*, data cabang, dan data barang. Sedangkan data transaksi meliputi data penjualan, data pembelian, dan data mutasi barang. Dalam database cabang hanya menyimpan data transaksi pada cabang tersebut dan data master, sedangkan dalam database pusat menyimpan seluruh data transaksi dan data master. *Syncronization app* bekerja pada masing-masing cabang, dan terhubung dengan database cabang yang digunakan oleh aplikasi PoS pada cabang tersebut. Sedangkan *web service* terhubung dengan database pusat, dan betugas melayani serta mengontrol proses sinkronisasi dari *syncronization app*. Struktur desain sistem ditunjukkan pada Gambar 1.



Gambar 2. *Flowchart* Sinkronisasi Database Menggunakan *Middleware*

Secara umum proses sinkronisasi dibagi menjadi dua bagian, yaitu proses pada *syncronization app* (SA) dan *web service* (WS) seperti yang ditunjukkan pada Gambar 2. Berikut langkah-langkah dalam proses sinkronisasi menggunakan *middleware*:

1. SA membaca waktu sinkronisasi terakhir, dan memeriksa database cabang apakah terdapat perubahan data setelah waktu sinkronisasi terakhir.
2. Jika terdapat perubahan data, SA akan memeriksa kondisi jaringan apakah kondisi jaringan sedang *online* atau *offline*.
3. Jika kondisi jaringan sedang *online*, SA mengirimkan perubahan data ke WS dan waktu terakhir sinkronisasi.
4. WS menerima perubahan data dari SA, dan membacanya satu persatu.
5. WS memeriksa jenis data, jika data termasuk data transaksi, maka WS langsung melakukan update ke database pusat, dikarenakan data transaksi pada masing-masing cabang berbeda satu sama lain, sehingga diasumsikan tidak akan terjadi konflik data.
6. Jika data termasuk data master, maka akan dilakukan pemeriksaan pada database pusat, apakah data tersebut sudah ada.
7. Jika data belum ada pada database pusat, maka WS akan menambahkan data tersebut.
8. Jika data sudah ada, akan dilakukan pemeriksaan data menggunakan *data marker*. Jika data yang dikrim SA lebih baru dari data dalam database pusat, WS akan memperbaharui data pada database pusat dengan data yang diterima dari SA.
9. WS mengambil data master dari database pusat, yang mengalami perubahan setelah waktu terakhir sinkronisasi SA.
10. Data tersebut dijadikan balasan ke SA.
11. SA menerima balasan dari WS, jika dalam balasan tersebut terdapat data, maka SA akan melakukan *update* data tersebut pada database lokal.
    SA memperbaharui waktu terakhir sinkronisasi.

6. Hasil dan Pembahasan

Pada bagian ini ditampilkan hasil dan pembahasan mengenai penerapan proses sinkronisasi menggunakan *middleware* pada butik RAIN. Gambar 3 menunjukkan tampilan *control panel* yang terhubung dengan database pusat. Dengan adanya *control panel* tersebut, status sinkronisasi pada masing-masing cabang dapat di-*monitoring* dan dapat digunakan untuk melihat data pada masing-masing cabang seperti data ketersediaan barang, data penjualan, data pembelian, dan data mutasi. Sehingga, untuk melihat ketersediaan barang pada semua cabang dapat dilakukan dengan lebih mudah dan cepat dibandingkan harus melakukan koneksi pada masing-masing cabang. Hal tersebut dibuktikan oleh hasil percobaan yang telah dilakukan, seperti yang ditunjukkan pada Tabel 1.

Gambar 3. *Dashboard Control Panel*

Percobaan berupa perbandingan dilakukan untuk melihat perbedaan antara perkembangan sebelum dan setelah memanfaatkan *middleware* dalam proses sinkronisasi database. Percobaan ini dilakukan dengan menggunakan perbandingan waktu yang diperlukan dalam melakukan pengecekan ketersediaan barang sebelum dan setelah penerapan sinkronisasi database menggunakan *middleware*. Perbandingan waktu yang kami uji adalah berapa lama waktu yang dibutuhkan untuk melakukan akses database hingga menampilkan hasil yang diminta pada aplikasi PoS tersebut. Akses data yang digunakan yaitu data ketersediaan barang pada setiap cabang, terlihat bahwa perbandingan waktu sangat jauh berbeda antara sebelum memanfaatkan *middleware* dan setelah menggunakan

middleware. Jika sebelum adanya pemanfaatan *middleware* pengguna harus melakukan akses data satu per satu pada setiap database yang dimiliki oleh cabang butik, dan pengguna harus melakukan sinkronisasi database secara manual dengan melakukan *import* dan *export* database. Maka saat ini pengguna hanya perlu melakukan akses data yang didapatkan langsung dari database pusat setalah adanya sinkronisasi database yang dilakukan oleh *middleware*.

Tabel 1. Hasil percobaan

Percobaan Ke	***lead time* (ms)**										
	Middle-ware	RAIN BATIK	RAIN VB	RAIN TZ	RAIN 4	RAIN Online	M Clothing	SUI	Nikkou PTC 1	Nikkou PTC 2	Nikkou Loop
1	233	16782	18364	3700	8151	13359	517	11442	13277	19202	8872
2	1036	18010	7381	3500	27457	13342	608	16745	19085	3618	16684
3	1011	4004	15119	17510	19462	17382	1213	3494	17520	3587	15447
4	953	12386	3362	3300	16498	16256	697	18774	17018	3370	13756
5	421	11114	10456	3234	3579	440	418	19991	10941	5838	3630
6	456	11091	13271	14257	11051	3176	18050	11121	7094	3662	15987
7	257	4003	3001	14821	3219	8251	29913	5112	3602	17114	3411
8	427	4004	3000	3001	17978	3288	11266	3327	13885	3556	5835
9	9279	4004	9337	9044	3510	9158	639	11341	13165	3252	16580
10	452	15523	3635	7381	3268	21127	6134	9333	10321	7483	16271
11	333	12741	3220	16467	15550	17362	398	17428	15247	14362	8390
12	379	10935	8814	7408	9935	3506	7344	9680	6545	5511	9300
13	1041	8298	7151	13237	9195	9406	387	9675	3414	3494	10894
14	3494	7808	6552	3335	3975	659	374	8652	6578	10978	11500
15	784	12247	3476	3333	3286	7089	394	3354	8897	3734	8707
16	612	4005	13427	3481	9675	7514	615	7444	12310	3581	7121
17	2567	8128	3347	4312	7179	8719	1237	7131	10919	3954	7899
18	9293	11259	7985	4051	7797	8578	710	3638	8047	3520	8279
19	831	10596	8991	3576	7409	9143	648	7696	9997	3438	9975
20	1224	9698	12251	4056	18267	19623	562	6525	10625	3458	3406
	Rata-rata										
ms	1754,1	9831,8	8107	7150,2	10322	9868,9	4106,2	9595,1	10924,5	6335,6	10097,2
s	1,754	9,832	8,107	7,150	10,322	9,869	4,106	9,595	10,924	6,336	10,097

Rata-rata waktu yang dihasilkan ketika melakukan akses secara langsung dari setiap cabang yaitu 8.2 detik dibandingkan dengan ketika melakukan akses secara langsung setelah pemanfaatan *middleware* yaitu 1.8 detik. Perbedaan ini tentu mungkin tidak akan terlalu dirasakan oleh pengguna apabila pengguna melakukan cek *stock* pada satu cabang butik, namun berbedaan ini akan dirasakan apabila pengguna melakukan cek *stock* pada seluruh cabang. Dalam penerapan sebuah teknologi adanya *lead time* ini merupakan salah satu hal yang seharusnya dapat diminimalkan keberadaannya karena semakin banyaknya waktu yang dihabiskan hanya untuk melakukan akses data dapat mengganggu jalannya kegiatan operasional lainnya. *Lead time* merupakan jangka waktu yang dibutuhkan sebuah aplikasi yang dihitung sejak pengguna melakukan *request* data hingga menerima data tersebut. Dapat dibayangkan jika nantinya butik ini memiliki penambahan cabang, maka akan semakin besar perbedaan *lead time*, karena semakin banyak datatabase yang harus diakses. Dengan pemanfaatan *middleware*, pengguna dapat mengumpulkan data pada masing-masing cabang secara terpusat sehingga pengguna hanya perlu mengakses pada satu tempat.

7. Kesimpulan

Dalam penelitian ini kami melakukan optimasi aplikasi *point of sale* (PoS) dengan cara memanfaatkan sinkronisasi data menggunakan *middleware*. Salah satu karakter *middleware* yang harus diperhatikan adalah kemampuannya untuk dapat diukur. Yang dimaksud dapat diukur adalah perbandingan berapa banyak informasi yang di *request* dengan informasi yang dapat diproses. Hal tersebut berkaitan dengan optimasi yang dilakukan pada PoS butik RAIN dimana sinkronisasi dapat dikatakan lebih efisien ketika pengguna tidak perlu lagi melakukan update data secara manual ketika akhir jam kerja. Namun update data dapat dilakukan secara otomatis ketika

syncronization app mendeteksi perubahan data setelah sinkronisasi terakhir. Jika terdapat perubahan data dan jaringan *online*, maka perubahan tersebut akan langsung dikirim ke *web service.* Akan tetapi, jika jaringan *offline*, maka pengiriman data ditunda jaringan *online*. Hasil pengujian pada penilitian ini dapat dilihat dari rata-rata waktu akses yang lebih kecil dibandingkan rata-rata waktu akses sebelum adanya *middleware*. Kelebihan lain yang dirasakan dari aplikasi tersebut adalah kemampuannya yang dapat diakses kapan saja (dalam kondisi *online* maupun *offline*) tanpa adanya gangguan dari koneksi yang dialami. Sehingga dapat disimpulkan bahwa aplikasi *middleware* tersebut dapat menjadi solusi yang tepat untuk meningkatkan produktivitas dan efektivitas kerja butik RAIN.

Referensi

[1] Aznoli, Fariba., dan Nima Jafari Navimipour. *Cloud Service Recommendation: Reviewing the Recent Advances and Suggesting the Future Research Directions*. Journal of Network and Computer Application 77, pp. 73-86, 2017

[2] Gartner Inc., Worldwide Public Cloud Services Market is Forecast to Reach $204 Billion in 2016, Gartner Inc. 2016. URL: http://www.gartner.com/

[3] Shodiq, Muhsin., Rini Wongso, Rendy Setya Pratama, EkoRhenardo, dan Kevin. *Implementation of Data Synchronization with Data Marker Using Web Service Data*. Procedia Computer Science 59, pp. 366-372, 2015

[4] Kaur, A., Aashima. *Synchronized Algorithm for Database and Image Processing Between Client and Server*. International Journal of Computer Science and Information Technologies, 2014

[5] Isak Shabani, dkk. *Solving Problems in Software Applications through Data Synchronization in Case of Absence of the Network.* IJCSI International Journal of Computer Science Issues Vol. 9, Issue 1, No 3, pp. 1694-0814, 2012

[6] Sutanto, Yusuf. dan Riyanarto Sarno. *Inventory Management Optimization Model with Database Synchronization through Internet Network (A Simulation Study)*. The 5th International Conference on Electrical Engineering and Informatics, 2015

[7] Wang, Jue. dan Dong-Song Zhang. *Research and Design of Distributed Database Synchronization System Based on Middleware*. The 8th International Conference on Intelligent Computation Technology and Automation., 2015

[8] Malhotra, N., Chaudhary, A. *Implementation of Database Synchronization Technique between Client and Server*. International Journal of Engineering Science and Innovative Technology, 2014

[9] Y. Lin, B. Kemme, Patino-Martinez, dkk. *Middleware based data replication providing snapshot isolation*. Proceedings of SIGMOD, pp. 419-430, 2005

[10] B. Kemme and G. Alonso. *Database replication: a tale of research across communities*. Proceedings of VLDB, pp. 5-12, 2010

[11] Vinoski, Steve. *An Overview of Middleware*. Springer-Verlag Berlin Heidelberg, pp. 35–51, 2004

[12] Ko Lagerberg, Dirk-Jaap Plas, and Maarten Wegdam. *Web services in Third-Generation Service Platforms.* Bell Labs Technical Journal 7 (2), pp. 167–183, 2002